Linear Algebra and its Applications 431 (2009) 318–333

Contents lists available at ScienceDirect

Linear Algebra and its Applications

journal homepage: www.elsevier.com/locate/laa

Exploiting structure in large-scale electrical circuit and power system problems

Joost Rommes [a,*,1], Nelson Martins [b]

[a] NXP Semiconductors, Corporate I&T/DTF, HTC 46 2.10, 5656 AE Eindhoven, The Netherlands
[b] CEPEL, P.O. Box 68007, Rio de Janeiro, CEP-21944-970, Brazil

ARTICLE INFO

Article history:
Received 15 September 2008
Accepted 29 December 2008
Available online 6 February 2009

Submitted A. Wathen

This paper is dedicated to Henk van der Vorst, on the occasion of his 65th Birthday.

Keywords:
Eigenvalues
Large-scale eigenvalue problems
Poles
Small-signal stability
Model order reduction
Arnoldi method
Jacobi–Davidson method

ABSTRACT

The rapid increase in complexity of systems such as electrical circuits and power systems calls for the development of efficient numerical methods. In many cases, direct application of standardized methods for numerical problems is computationally not feasible or inefficient. However, the performance of such methods can be improved considerably by taking into account the structure of the underlying problem. In this paper, we describe when and how this – mathematical and/or physical – structure can be exploited to arrive at efficient algorithms that also suffer less from other numerical issues such as round-off errors. Eigenvalue and stability problems are considered in particular, but applications to other problems are shown as well.

1. Introduction

Complex systems such as electrical circuits and power systems are usually modeled by systems of differential-algebraic equations. Simulation in time or frequency domain of the resulting dynamical systems heavily relies on numerical methods for eigenvalue problems, systems of linear equations, time integration, and model order reduction [3,5,31]. The rapid increase in complexity of these systems, due to the continuous addition of functionality on both smaller and larger scale, calls for the development

* Corresponding author.
E-mail addresses: joost.rommes@nxp.com (J. Rommes), nelson@cepel.br (N. Martins).
URL: http://rommes.googlepages.com (J. Rommes), http://www.nelsonmartins.com (N. Martins).
[1] The first author was supported by EU Marie-Curie Project O-MOORE-NICE!

0024-3795/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.laa.2008.12.027

of efficient numerical methods: since the number of unknowns or states in the systems varies from a few thousands to millions, direct application of standardized (but expensive) numerical methods, that for historical reasons (e.g., robustness) are often used in practice, is computationally not feasible or highly inefficient.

It is not only size that matters. Eigenvalue problems can be solved completely by full space methods such as the QR method [14] on laptop computers for up to a few thousands unknowns, and partially by iterative methods such as the (implicitly restarted) Arnoldi method [34] or the Jacobi–Davidson method [32] for even much larger systems. In several applications and especially in stability analysis, where one is interested in the rightmost eigenvalues, the presence of eigenvalues at infinity may disturb the computational process, possibly leading to wrong conclusions about stability [22,27]. Similar problems also cause difficulties in the reduction of dynamical systems with balancing transformations [35,36].

In this paper, we describe when and how the structure of the underlying problem can be exploited to come to efficient algorithms for large-scale problems that also suffer less from numerical issues such as round-off errors. We give a survey of methods that exploit structure in various problems related to dynamical systems, for stability analysis in particular and model order reduction in general. Of course, not in all situations one is able to exploit the structure: the structure may be hidden or even unknown if it is not exactly clear what the underlying problem is. Fortunately, there are applications, for instance in circuit and power system simulation, where the structure is more or less explicitly available. If that is the case, techniques described in this paper can be used to make execution of (existing) numerical methods faster and more robust.

This paper is organized as follows. In Section 2, we motivate the need for specialized methods by describing the modeling and analysis of power systems. In Section 3, we show how in various applications the structure of the underlying problem can be exploited to come to efficient algorithms. We illustrate the benefits of this in Section 4 by examples from practice. Section 5 concludes.

2. Motivation and examples from practice

In this section, we describe in more detail the modeling of linearized power system stability problems as it is done in practice. As will become clear, this modeling leads to structured dynamical systems – it is this structure that we will exploit in the following sections. We stress that similar structures also arise in the modeling of electrical circuits [8], in fluid dynamics [9,22], and in structural dynamics [38]. The numerical examples that we describe in Section 4 are created following the modeling described in this section.

2.1. Power systems

The power system electromechanical stability problem can be represented by a set of differential equations together with a set of algebraic equations [16,20]:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{z}), \\ 0 = \mathbf{g}(\mathbf{x}, \mathbf{z}), \end{cases} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{n_s}$ is the state vector, $\mathbf{z} \in \mathbb{R}^{n_z}$ is a vector of algebraic equations, and $\mathbf{f} \in \mathbb{R}^{n_s} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_s}$ and $\mathbf{g} \in \mathbb{R}^{n_s} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ describe the system dynamics and relations. In the following $n = n_s + n_z$, and in general we have $n_z \gg n_s$.

In the case of small-signal stability analysis, system (1) is linearized around a system operating point $(\mathbf{x}_0, \mathbf{z}_0)$ to yield

$$\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\dot{\mathbf{x}} \\ 0 \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{z} \end{bmatrix}, \tag{2}$$

where J_i $(i = 1, \ldots, 4)$ of appropriate dimensions form the Jacobian

$$A = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix}$$

of the system and Δ denotes an incremental change in steady-state value (in the following we will omit Δ for notational simplicity). Note that the matrices J_i are sparse in general. System (2) is in so-called

descriptor form, also known as augmented Jacobian form in power system engineering [20]. It is well known that the small-signal stability of (1) in the neighborhood of operating point $(\mathbf{x}_0, \mathbf{z}_0)$ can be analyzed by inspecting the eigenvalues of the pencil (A, E), where $E = \text{blkdiag}(I, 0)$: if there are finite eigenvalues with positive real part, the system is unstable.

If in addition to small-signal stability there is interest in the input–output behavior of the linearized system, one studies linear time invariant systems of the form

$$\Sigma_J \equiv \begin{cases} \dot{\mathbf{x}}(t) = J_1\mathbf{x}(t) + J_2\mathbf{z}(t) + B_1\mathbf{u}(t), \\ 0 = J_3\mathbf{x}(t) + J_4\mathbf{z}(t) + B_2\mathbf{u}(t), \\ \mathbf{y}(t) = C_1^T\mathbf{x}(t) + C_2^T\mathbf{z}(t) + D\mathbf{u}(t), \end{cases} \tag{3}$$

where $\mathbf{u}(t) \in \mathbb{R}^m$ and $\mathbf{y}(t) \in \mathbb{R}^p$ are the input and output, respectively, and the matrices $B = [B_1^T, B_2^T]^T \in \mathbb{R}^{n\times m}$ and $C = [C_1^T, C_2^T]^T \in \mathbb{R}^{n\times p}$ are input-to-state and state-to-output mappings, respectively. The matrix $D \in \mathbb{R}^{p\times m}$ is the direct transmission map. Throughout this paper we will also refer to the more general descriptor form

$$\Sigma_J \equiv \begin{cases} E\dot{\mathbf{x}}_d(t) = A\mathbf{x}_d(t) + B\mathbf{u}(t), \\ \mathbf{y}(t) = C^T\mathbf{x}_d(t) + D\mathbf{u}(t), \end{cases} \tag{4}$$

where the dimensions of (E, A, B, C, D) can be deduced readily from (3) and $\mathbf{x}_d = [\mathbf{x}^T, \mathbf{z}^T]^T$.

In power system stability analysis and modeling the matrix J_4 is nonsingular [20]. This allows to eliminate the algebraic variables $\mathbf{z}$ from the system and to reduce the system to state space form

$$\Sigma_{ss} \equiv \begin{cases} \dot{\mathbf{x}}(t) = A_s\mathbf{x}(t) + B_s\mathbf{u}(t), \\ \mathbf{y}(t) = C_s^T\mathbf{x}(t) + D_s\mathbf{u}(t), \end{cases} \tag{5}$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_s}$ is the state vector, $\mathbf{u}(t) \in \mathbb{R}^m$ the input vector and $\mathbf{y}(t) \in \mathbb{R}^p$ the output vector; $A_s \in \mathbb{R}^{n_s\times n_s}$ is the state space matrix, and $B_s \in \mathbb{R}^{n_s\times m}$, $C_s \in \mathbb{R}^{n_s\times p}$ and $D_s \in \mathbb{R}^{p\times m}$ are system matrices. The transfer function of system (5) is defined as $H(s) = C_s^T(sI - A_s)^{-1}B_s + D_s$. The frequency response of $H(s)$ is obtained by computing $H(s)$ for discrete values of $s = j\omega_i$ along a frequency interval $(\omega_1, \ldots, \omega_k)$.

The state space (5) and descriptor representations (2) are related as follows:

$$\begin{aligned} &A_s = J_1 - J_2J_4^{-1}J_3, \quad B_s = B_1 - J_2J_4^{-1}B_2, \\ &C_s^T = C_1^T - C_2^TJ_4^{-1}J_3, \quad D_s = D - C_2^TJ_4^{-1}B_2. \end{aligned} \tag{6}$$

The differential-algebraic equations (DAEs) in (4) are usually assembled per individual power system component, for easy bookkeeping (see Fig. 1, where we refer to the blocks J_a, J_b, J_c, and J_d to stress the difference with (3)), rather than strictly keeping the state equations (J_1 and J_2) separate from the algebraic equations (J_3 and J_4), as indicated in (3). The modeled components are synchronous generators and associated controllers (terminal voltage regulators, prime-movers and rotor-speed regulators controllers, oscillation damping controllers – widely known as power system stabilizers). Block J_a is comprised by the differential-algebraic equations for synchronous machines, as well as by induction motor loads, high-voltage DC transmission links (HVDC, used for very long distance transmission), high-power electronic equipment for improved dynamic performance (FACTS – flexible AC transmission systems), wind power generators, etc. All the equipment that are relevant to the study in hand, are interconnected through a network of extra-high-voltage transmission lines (the higher the voltage, the less are the resistive losses incurred in transferring electric power from distant generating plants to major load centers). This network consists of lines of different voltage levels, which are interconnected by power transformers and usually have many shunt capacitors and reactors for voltage control. The network has intrinsic dynamics (of electromagnetic nature) that are much faster than that associated with the transient behaviors of the various interconnected machines and controllers, which are mainly of electromechanical nature. The electrical network modeling, which comprises matrix J_d, is therefore done by algebraic equations only, reflecting the relatively instantaneous response of the network. The matrices J_b and J_c cater for the (also algebraic) interconnection of equipment in J_a with the network (in J_d).

Matrix A is sparse because the electrical network is sparse: the number of nonzero elements per nodal equation is equal to twice the number of transmission lines connected to that electrical node

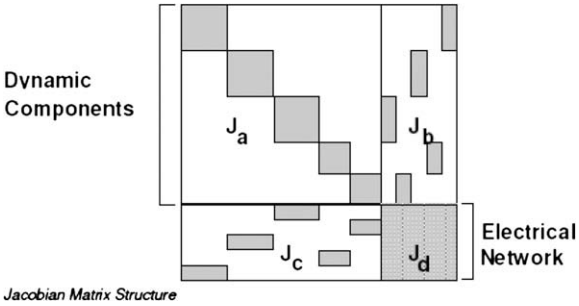

Fig. 1. Structure of the Jacobian matrix in power system modeling.

(in practice, a node is a substation or part of a substation) plus one. In practical power systems, the average value for connected transmission lines is between two and three per node, at maximum. Only a few of the equations for the individually modeled dynamic components are directly connected to the algebraic equations of the electrical network, as either current injections into the node (nonzero elements in J_c) or as variables directly disturbed by the voltage deviations in a given node (nonzero elements in J_b). The same J_d nodal matrix structure is used to efficiently solve steady-state power flow conditions and short-circuit (different types of faults) computations, to determine if a system design is adequate for heavy load, fault as well as permanent line-outage conditions. For more details about power system modeling, see for instance [16].

A typical practical problem in power system dynamics and control engineering is the configuration of Power System Stabilizers (PSS) [10]. In this process a number of machines should have their PSS gains properly coordinated. The effect on the stability of the system can be traced by considering root-locus plots showing the eigenvalues for different values of their gains; see Fig. 2 for an example. The goal of the tuning process is to get all the eigenvalues in the left half-plane beyond a minimum damping ratio level,[2] since in that case the system is stable and resilient. The root-locus traces in Fig. 2 are computed using the Sensitive Pole Algorithm [28], an eigensolution method that computes the eigenvalues that are most sensitive to changes in the matrix. Besides the eigenvalues most sensitive to changes in damping controller parameters of critical generating plants, one is also interested in the rightmost eigenvalues, since these determine the (in)stability of the system. In the following section we will discuss algorithms to compute the rightmost eigenvalues in an efficient way.

3. Exploiting structure in dynamical systems

In this section, we discuss computational aspects of working with state space and descriptor matrices, and we show how one can take both the numerical advantages of state space matrices and the computational advantages of descriptor matrices at the same time. We briefly review techniques for the computation of rightmost eigenvalues in the presence of eigenvalues at infinity [22,27], and show how these techniques can be simplified considerably in special cases that typically occur in electrical and power system modeling. Furthermore, we describe how similar techniques can be used for the solution of large matrix equations and model order reduction.

3.1. Sparse descriptor form vs. state space form

As described in Section 2.1, the dynamical behavior of power systems can be described by a set of differential and algebraic equations (1). In the case of small-signal stability analysis, system (1) is linearized around a system operating point $(\mathbf{x}_0, \mathbf{z}_0)$ to yield (2).

[2] The damping ratio ζ of an eigenvalue $\alpha \pm \beta i$ is defined as $\zeta = \frac{-\alpha}{\sqrt{\alpha^2+\beta^2}}$.

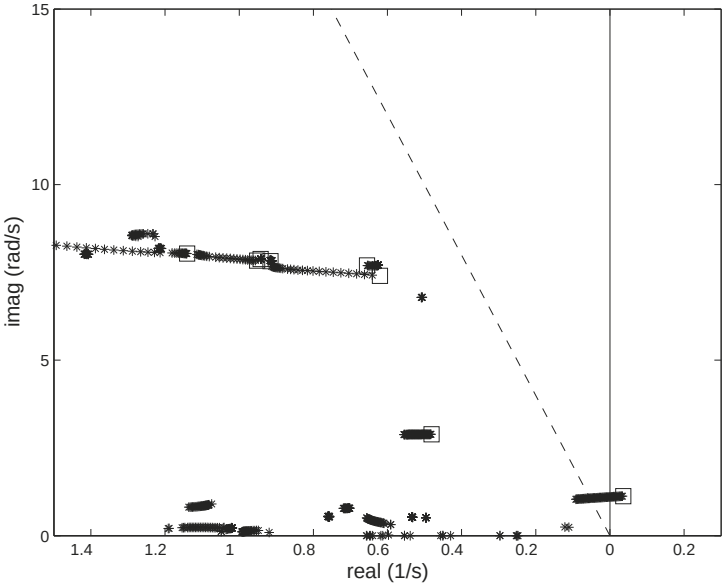

Fig. 2. Root-locus traces of eigenvalues most sensitive to changes in the gains of the power system stabilizers (computed using the Sensitive Pole Algorithm [28]). Additionally, power system engineers are interested in eigenvalues with low damping ratios (the dashed line indicates the 5% damping ratio boundary) and in the right half-plane in particular.

In general the number of algebraic variables n_z is (much) larger than the number of state variables n_s: in the example in Section 4, for instance, $n_z = 11{,}586$ while $n_s = 1664$. Eliminating the algebraic variables, hence, would mean a reduction of almost 90% of the variables. Another advantage is that together with the algebraic variables, also the infinite modes are eliminated from the system (these infinite modes are a possible source of numerical problems, as we will see later). Furthermore, for eigenvalue problems in practice, the QR method was preferred for being faster and more reliable than the QZ method and partial eigensolution methods. For these reasons, eliminating algebraic variables has been applied for many years in the development of algorithms for small-signal analysis of mid-sized power systems [20].

However, there is one major drawback connected to the elimination of algebraic variables: while the original Jacobian A is very sparse, the resulting state space matrix A_s after elimination is dense. This has severe consequences for the performance of algorithms for eigenvalue problems and model order reduction. Since the main operation in such algorithms is the (repetitive) solution of linear systems of the form $(A_s + \sigma I)\mathbf{x}_s = \mathbf{b}_s$ (in state space form) or $(A + \sigma E)\mathbf{x} = \mathbf{b}$ (in descriptor form) for shifts $\sigma \in \mathbb{C}$, sparsity of the corresponding matrix A_s (or E and A in descriptor form) is of crucial importance. By making use of row and column permutations such as AMD [2], the costs for solving linear systems in case of sparse descriptor matrices related to power and electrical systems is often $O(n^\alpha)$, where $1 \leqslant \alpha \leqslant 2$ [26]. For dense system matrices, on the other hand, one usually faces the traditional costs of $O(n^3)$. Since the state space matrix is usually dense, it is easy to see that even with a reduction of 90% of the variables, the sparse descriptor formulation is still favored for large-scale power systems (see also Section 4).

To get an idea of the fill-in generated by eliminating the algebraic variables, consider the spy plots in Fig. 3. For larger systems the number of nonzero elements in the state space matrix soon becomes excessive, and the gain in speed (and memory) when solving linear systems through use of the sparse Jacobian is easily above 90% (see also Section 4).

The main message of this section is that from a computational viewpoint, one should use the sparse system matrices, if available, as much as possible, and refrain from eliminating algebraic variables

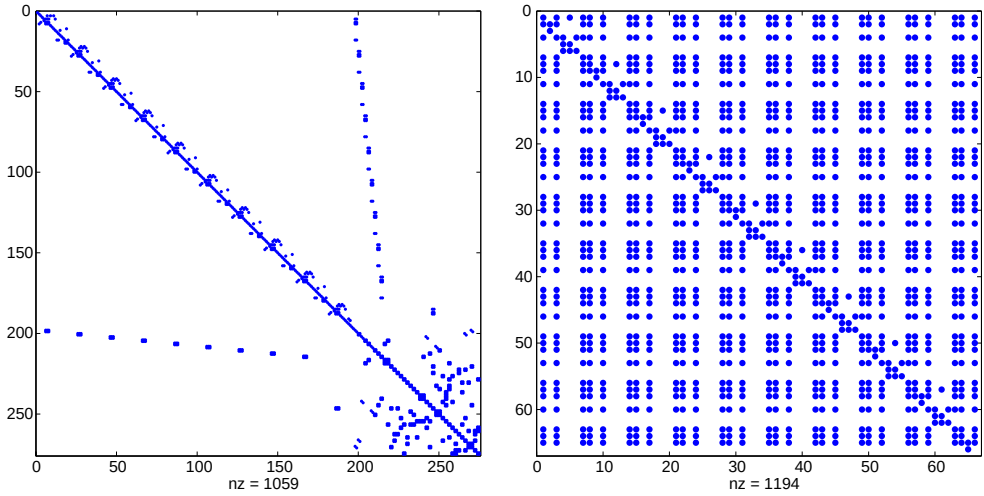

Fig. 3. Sparse Jacobian matrix (left) and dense state space matrix of 66-state power system model.

explicitly.[3] In the following sections we will show how one can still profit from the other advantage of the state space formulation (elimination of infinite modes) – in an implicit way.

3.2. Implicit operations with $A_s = J_1 - J_2 J_4^{-1} J_3$

In iterative methods for solving eigenvalue problems or Lyapunov equations one needs to compute matrix–vector (MV) products ($\mathbf{y} = (A_s + \sigma I)\mathbf{x}$ where $\sigma \in \mathbb{C}$ is a shift) and solve linear equations ($(A_s + \sigma I)\mathbf{x} = \mathbf{b}$). If A_s is dense, then MVs will be expensive and solves probably impossible. Now assume that E and A have the structure as described in Section 2, i.e.,

$$E = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix}, \tag{7}$$

where $J_4 \in \mathbb{R}^{k \times k}$ is nonsingular. As described in Section 2, we have the relation

$$A_s = J_1 - J_2 J_4^{-1} J_3.$$

In general it is not practical to construct A_s explicitly, since it will be dense. Instead, it is more economical to implement MVs and solves with A_s implicitly. Algorithm 1 shows a rather straightforward implementation of the matrix–vector product $\mathbf{y} = (A_s + \sigma I)\mathbf{x}$ where σ is a shift. These concepts are used in power system simulation software [20]. Note that this approach is profitable if the number of MVs and solves is less than n_s^2, which is in general the case for iterative methods.

Algorithm 1: Matrix–vector product $\mathbf{y} = (A_s + \sigma I)\mathbf{x}$

INPUT: Descriptor matrices J_1, J_2, J_3, J_4, shift σ, vector $\mathbf{x}$
OUTPUT: $\mathbf{y} = (A_s + \sigma I)\mathbf{x}$
1: Compute $\mathbf{z} = J_3\mathbf{x}$
2: Solve $\mathbf{t}$ from $J_4\mathbf{t} = \mathbf{z}$
3: Compute $\mathbf{y} = J_1\mathbf{x} - J_2\mathbf{t} + \sigma\mathbf{x}$

For implicitly solving systems of the form $(A_s + \sigma I)\mathbf{x} = \mathbf{b}$ with $A_s = J_1 - J_2 J_4^{-1} J_3$ (A_s is the Schur complement of J_4 in A) we note that [12,15]

[3] For small and moderate scale problems working with the dense state space matrices can be more efficient than working with the sparse descriptor matrices. Since the number of state space variables is often known, one can determine a priori whether to work with the state space or descriptor matrices.

$$\begin{bmatrix} J_1+\sigma I & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ * \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix} \Longleftrightarrow (A_s+\sigma I)\mathbf{x} = \mathbf{b}.$$

Solves with the sparse descriptor matrices are in general much cheaper than (construction of and) solves with the dense state space matrices. An implementation is shown in Algorithm 2.

Algorithm 2: Solve $\mathbf{x}$ from $(A_s+\sigma I)\mathbf{x} = \mathbf{b}$

INPUT: Descriptor matrices J_1, J_2, J_3, J_4, shift s, vector $\mathbf{b}$
OUTPUT: $\mathbf{x} = (A_s+\sigma I)^{-1}\mathbf{b}$
1: Solve $\mathbf{t} = [\mathbf{x}^T, \mathbf{z}^T]^T$ from

$$\begin{bmatrix} J_1+\sigma I & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ 0 \end{bmatrix}$$

2: Extract $\mathbf{x}$ from $\mathbf{t}$

A natural question is why one should work with A_s (implicitly), while the descriptor matrices are explicitly available. This will become clear in the following sections, where we will show how these implicit multiplications and solves can be used in algorithms for computing eigenvalues and solutions of Lyapunov equations.

3.3. Eigenvalue problems

Stability and vibration analyses are important to a wide range of application areas, including power system dynamics, electrical circuit simulation, fluid dynamics, structural engineering and geophysics. The problem of computing the rightmost eigenvalues of large matrices or matrix pencils is notorious for the numerical difficulties that might be connected to the presence of eigenvalues at infinity. In fluid dynamics, in particular in the stability analysis of steady-state solutions of the Navier–Stokes equations, the appearance of spurious or ghost eigenvalue approximations is well known [9,22,27]. This problem is also observed in stability and pole-zero analysis of electrical circuits [7]. Purification techniques [22,27] and alternatives [6] have been developed to prevent iterative methods such as implicitly restarted Arnoldi and Jacobi–Davidson to approximate these eigenvalues at infinity. Successful application of these methods depends on the severity of the problem (i.e., the size of Jordan blocks corresponding to the eigenvalues at infinity) and the use of clever shifts for (modified) Cayley transformations.

Here we show that under certain circumstances the purification procedure can be simplified considerably. We are interested in the rightmost eigenvalues of the pencil $(A, E) \in (\mathbb{R}^{n\times n}, \mathbb{R}^{n\times n})$, i.e., we are looking for the finite λ with largest real part that satisfy

$$A\mathbf{x} = \lambda E\mathbf{x}, \quad \mathbf{x} \neq 0,$$

where $\mathbf{x}$ is an eigenvector. Assume that A and E have the structure as described in Section 2, i.e.,

$$A = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \quad \text{and} \quad E = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \tag{8}$$

where $J_4 \in \mathbb{R}^{k\times k}$ is nonsingular. Note that if $(\lambda, \mathbf{x}_s)$ is an eigenpair of $A_s = J_1 - J_2 J_4^{-1} J_3$, then $(\lambda, [\mathbf{x}_s^T, -(J_4^{-1}J_3\mathbf{x}_s)^T]^T)$ is an eigenpair of (A, E), a property that is also used in other application areas [6,22,27]. There are several approaches to find the rightmost eigenvalues of (generalized) eigenvalue problems:

- Use Implicitly Restarted Arnoldi (IRA) [17] with Shift-and-Invert. Typically a shift $\sigma = 0$ is chosen, i.e., IRA is applied to $A^{-1}E$ (without computing A^{-1} explicitly, see Section 3.6). A difficulty here can be that the rightmost eigenvalues of (A, E) are not the eigenvalues with largest magnitude of $A^{-1}E$, and if the rightmost eigenvalue has negative real part, it may no longer be the rightmost eigenvalue of the shift-and-inverted matrix. There are several techniques to deal with this, see for instance [6,23,27].
- Use the Jacobi–Davidson QZ method [11]. The advantage of JDQZ is that we do not have to invert A and can work with the pencil (A, E). Furthermore, we can select the rightmost Petrov values

(eigenvalue approximations) at every iteration to increase the chances of finding indeed the rightmost eigenvalues. A possible disadvantage is that one may need a good preconditioner to solve the correction equations [11] for computing eigenvector updates (if direct solution is infeasible).

The main factor that disturbs the performance of the above approaches is the presence of n_z eigenvalues at infinity, with eigenspace

$$V_\infty = \begin{bmatrix} 0^{n_s \times n_z} \\ I^{n_z \times n_z} \end{bmatrix} \in \mathbb{R}^{n \times n_z}$$

As soon as the Krylov space in the Arnoldi process gets polluted by components in the direction of V_∞, Ritz values may approximate eigenvalues at infinity, possibly leading to wrong conclusions on the stability. Purification techniques [22,27] can be used to keep the Krylov space (or search space for Jacobi–Davidson) free from directions in V_∞.

The special structure of E and A in (8) together with the (reasonable) assumption that solves with J_4 and $sE - A$ are cheap, suggests the following approaches:

1. Use Arnoldi or Jacobi–Davidson to compute the rightmost eigenvalues of $A_s = J_1 - J_2 J_4^{-1} J_3$. Since all eigenvalues at infinity are eliminated, the process is not hampered by those eigenvalues. Because we use the state space matrix (implicitly) and provide as selection criterion "rightmost", we increase the chances of converging to the rightmost eigenvalues (contrary to when using a shift-and-invert approach, as discussed above). To avoid the explicit construction of A_s, we supply IRA (`eigs` in Matlab [37]) or Jacobi–Davidson with a routine for computing $\mathbf{y} = A_s\mathbf{x}$ following Algorithm 1. In the Jacobi–Davidson method one can use Algorithm 2 to solve the correction equations in an efficient way.
2. Use Arnoldi with Shift-and-Invert. Again, instead of working explicitly with A_s, we implement solves of $(A_s - sI)\mathbf{x} = \mathbf{b}$ following Algorithm 2. The possible drawback of the Shift-and-Invert approach remains that several (Cayley) shifts may be needed to find the rightmost eigenvalues [6,27].

Advantages of the above approaches are that all eigenvalues at infinity are eliminated and that the effective search space is the state space, which is much smaller than the complete descriptor space. Furthermore, for approach 1 there is a reduced risk of missing the rightmost eigenvalues. In summary, if is possible and computationally attractive to implicitly work with the state space matrices, this is the best to do since problems with eigenvalues at infinity are eliminated; if this is not possible, for theoretical or computational reasons, one has to resort to purification techniques. In Section 4, the different approaches are compared on performance for practical examples.

In full space methods like the QR and QZ method it is much more difficult to exploit the sparsity or structure of the system, since one of the crucial steps in these methods is the reduction to upper-Hessenberg form (which will most likely be dense). Computation of all eigenvalues becomes too expensive for systems with several thousands of states and hence it is more attractive to use specialized methods, if existent, that compute only the eigenvalues of interest. Such specialized eigensolution methods are usually iterative subspace methods and can take full advantage of the inexpensive solves with the sparse descriptor matrices.

For some specific eigenvalue problems, it is even not needed to work implicitly in state space to avoid hampered convergence due to eigenvalues at infinity. Example problems are the computation of dominant poles and eigenvalues most sensitive to parameter variations. A pole is called dominant if its corresponding (normalized) right and left eigenvectors $\mathbf{v}$ and $\mathbf{w}$ have large components in the direction of the output vector $\mathbf{c}$ and input vector $\mathbf{b}$ of the dynamical system, i.e., if $|(\mathbf{c}*\mathbf{x})(\mathbf{y}*\mathbf{b})|$ is large [21]. Sensitive eigenvalues are defined in a similar way [28]. For many realistic systems, the poles at infinity are neither dominant nor sensitive, i.e., $|(\mathbf{c}*\mathbf{x})(\mathbf{y}*\mathbf{b})| = 0$ for $\mathbf{x}$ and $\mathbf{y}$ corresponding to eigenvalues at infinity. The Dominant Pole Algorithm (DPA) [21] and the Sensitive Pole Algorithm [28] converge to the dominant and sensitive poles and avoid any non-dominant and non-sensitive poles,

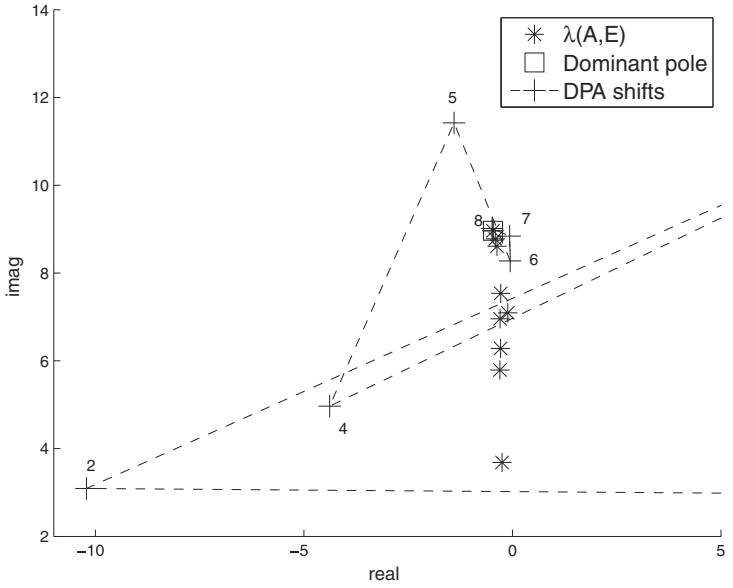

Fig. 4. Typical convergence of the Dominant Pole Algorithm. DPA starts with initial shift $\sigma = 10 + 5 * 10^3 i$ and converges to the most dominant finite pole of $H(s) = \mathbf{c}*(sI - A)^{-1}\mathbf{b}$, leaving apart less dominant poles and poles at infinity. The + symbols (and numbers) denote the iterates of DPA that, as indicated by the dashed line, converge to the most dominant pole (square). The DPA shifts at iteration 1 and 3 are outside the scope of the graph.

including those at infinity, as is analyzed in [30]. A nice illustration of this property is shown in Fig. 4, where starting from $10 + 5 \cdot 10^3 i$ DPA converges to the most dominant pole.

3.4. Model order reduction via balancing transformations

Model order reduction techniques for state space systems (A, B, C, D) (see (5), subscripts s are omitted for convenience) based on balanced truncation [24] require the solution of Lyapunov equations

$$AP + PA^T + BB^T = 0 \quad \text{and} \quad QA + A^T Q + CC^T = 0,$$

where $P \in \mathbb{R}^{n_s \times n_s}$ and $Q \in \mathbb{R}^{n_s \times n_s}$ are the controllability and observability gramians. In the past years many approaches have been developed to solve these equations for large systems as well. The most successful methods are based on computing the solutions P and Q as low-rank approximations with an ADI process [5,19,25].

These methods can be generalized to the descriptor case where $E \neq I$ is possibly nonsingular [35,36]. However, for this the computation of deflating subspaces corresponding to the finite eigenvalues of E is required, an operation that might be expensive in practice. Recently, two new approaches have been developed that take advantage of the structure of the underlying problem [12,15].

The idea here is similar to that described in the previous section: instead of working with dense state matrices, the sparse descriptor matrices are used during the solution of linear systems (note that in the ADI process, systems of the form $(A_s + \sigma I)\mathbf{x} = \mathbf{b}$ need to be solved at every iteration). One can use Algorithm 2 for this purpose. The low-rank approximations of P and Q, however, are still constructed in state space. In other words, we do not have to compute the deflating subspaces since we work in state space – implicitly when solving linear systems, and explicitly when constructing low-rank approximations. As an additional advantage, the sparse descriptor system solves are much cheaper than the dense state space system solves.

This approach has been applied successfully to model order reduction problems in fluid dynamics [15] and power systems [12].

3.5. Model order reduction via Krylov methods

Another recent advance in model order reduction is the development of structure preserving reduction techniques [5,13,31]. The idea is that the structure of the original system is preserved in the reduced-order model. As we have seen, many systems have a specific structure that is reflected in structured system matrices. The system matrices for models of RLCK electrical circuits (networks with resistors, inductors, capacitors, and mutual inductors) are of the form

$$E = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} G & P \\ -P^T & 0 \end{bmatrix},$$

where $C \in \mathbb{R}^{m\times m}$ and $G \in \mathbb{R}^{m\times m}$ contain the contributions of the capacitors and resistors, respectively, $L \in \mathbb{R}^{k\times k}$ contains the contributions of the (mutual) inductors, and P contains topological information related to the inductors. Projection based model order reduction techniques construct a basis $V \in \mathbb{R}^{n\times q}$, where $q \ll n$, for the dominant dynamics of the system, for instance using Krylov subspace methods [31]. The reduced-order model matrices are obtained as $E_q = V^T EV$ and $A_q = V^T AV$. It is clear that the original structure of the system is no longer present in the reduced model system matrices: the zero blocks are most likely replaced by dense matrices in the reduced system.

Recently developed methods [13,31] preserve the structure by projecting with

$$\widetilde{V} = \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix},$$

where $V_1 \in \mathbb{R}^{m\times q}$ and $V_2 \in \mathbb{R}^{k\times q}$. It is easy to see that now the block structure of the original system is preserved in the reduced model (at the cost of a twice as large model).

One can argue that this way only the structure at highest level is preserved. A closer inspection of the original system matrices shows that all nonzero blocks in E and A are sparse, while in the reduced model all nonzero blocks will be dense. Moreover, the topology matrix P is probably no longer a topology matrix (which contains only elements in $\{-1, 0, 1\}$): the reduced-order model can no longer be realized as a circuit. In industrial applications, however, realization of reduced-order models is important and current research focuses on methods that produce such models [40].

3.6. Other ways of exploiting structure

There are several other ways to exploit the – mathematical and/or physical – structure of the underlying problem. It is well known that products of (inverses of) sparse matrices should rarely be computed explicitly; instead, one uses a function that composes the product as the argument for an (generalized) eigenvalue solver or linear system solver.

A common technique for solving polynomial eigenvalue problems is to transform the polynomial eigenvalue problem into an equivalent generalized eigenvalue problem [38]. Although this might have advantages in some cases, it has the disadvantage that the dimensions of the matrices are multiplied by the order of the problem, e.g., a quadratic eigenvalue problem with $n \times n$ matrices is transformed into a generalized eigenvalue problem with $2n \times 2n$ matrices. In many cases it is advantageous to work directly with the polynomial formulation, see for instance [29,33].

Even closer to the originating problem is a concept that is used in some circuit simulators [1,8]. Large integrated circuits or chips are usually designed in a hierarchical way, i.e., the complete chip is composed out of several submodels, that are composed out of submodels as well. This allows for effective reuse of functionality and models in large chip designs: a simple component such as an amplifier might be used in several places on the chip and now needs to be designed only once. Advanced circuit simulators provide functionality that allows submodels to be defined only once, and to be instantiated as many times as needed. When simulating the design, mathematical methods can also take advantage of the present hierarchy (provided the hierarchy is known internally). A typical operation needed during time integration of a set of differential-algebraic equations, is the solution of large linear systems with Jacobian matrices. If the hierarchy is still reflected in the Jacobians, this automatically provides an ordering that allows efficient (LU or UL) factorization [8]. An even more advanced concept

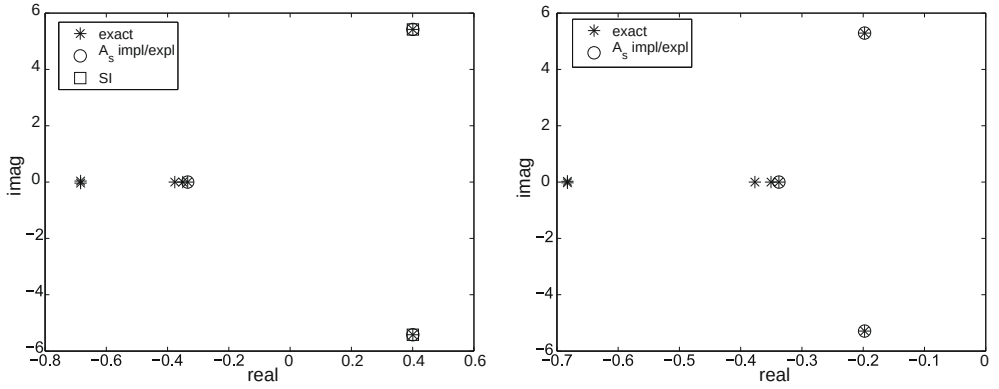

Fig. 5. Rightmost parts of the spectra for Kpss = 2 (left) and Kpss = 10. Arnoldi with target 'rightmost' converges to the two rightmost eigenvalues. Arnoldi with Shift-and-Invert only finds the rightmost eigenvalue (left) or fails (right).

is isomorphic matching [1], where one reuses (partial) solutions for a submodel for other occurrences of that submodel as well.

Another way of taking advantage of the structure of the problem is described in [4]. Here the matrices are so-called hierarchical matrices. It turns out that this specific structure can be exploited in for instance the solution of linear systems with hierarchical matrices as operator. In [4] this has led to efficient solvers for Lyapunov equations.

4. Numerical results

To illustrate the possible benefits of exploiting structure, we compare the performance of two methods for eigenvalue problems, the Arnoldi [34] and Jacobi–Davidson [32] methods to compute the rightmost eigenvalues in three scenarios: (1) explicitly apply the methods to the state space matrices, (2) explicitly apply the methods to the descriptor matrices, and (3) implicitly apply the methods to the state space matrices by working with the descriptor matrices.

All experiments were carried out in Matlab. For the implicitly restarted Arnoldi method (IRA), the Matlab implementation `eigs` was used (which is in fact a wrapper around ARPACK [18]), with Krylov space dimension $k = 20$. For the Jacobi–Davidson method a variant using real arithmetic was used [39]. Tolerance in both methods was $\tau = 10^{-6}$.

4.1. A small example

We first consider a small model of a 5-machine power system. The descriptor realization of this 41-state system has dimension 136. Two values for the power system stabilizer gain Kpss [10] are considered: for Kpss = 2 there is a complex-conjugated pair of eigenvalues with positive real part: $\lambda \approx 0.4 \pm 5.4i$. For Kpss = 10 the rightmost eigenvalues are $-0.2 \pm 5.3i$. The Arnoldi method is used to compute the two rightmost eigenvalues (here complex-conjugated pairs are counted as a single eigenvalue). The results are shown in Tables 1 and 2 (for this small system the computational costs for state space and descriptor computations are similar and hence not shown – we focus on the numerical properties instead). Fig. 5 shows (parts of) the spectra and the eigenvalues found by the different variants of Arnoldi.

We can make a couple of observations from Table 1, where the results are shown for a system with one eigenvalue (complex-conjugated pair) in the right half-plane: firstly, targeting the rightmost eigenvalues for the state space matrix A_s works best: the two rightmost eigenvalues are found, both in the explicit and the implicit case (note the number of iterations is the same since only the way we compute MVs with A_s changes, not the MV itself). Secondly, Shift-and-Invert (SI A_s with $\sigma = 0$) finds the rightmost eigenvalue, but fails to find the second-rightmost eigenvalue. Here there was no difference

Table 1
Results for Implicitly Restarted Arnoldi applied to $A_s\mathbf{x} = \lambda\mathbf{x}$ for computing two rightmost eigenvalues. In this case the rightmost eigenvalue was $\lambda = 0.4 \pm 5.4i$.

Target	#Restarts	Remarks
Rightmost A_s	144	Found two rightmost eigenvalues
SI A_s with $\sigma = 0$	8	Found rightmost eigenvalue
SI $A - \sigma E$ with $\sigma = 0$	300+	No results
Rightmost A_s via Algorithm 1	144	Found two rightmost eigenvalues

Table 2
Results for Implicitly Restarted Arnoldi applied to $A_s\mathbf{x} = \lambda\mathbf{x}$ for computing two rightmost eigenvalues. In this case the rightmost eigenvalue was $\lambda = -0.2 \pm 5.3i$.

Target	# Restarts	Remarks
Rightmost A_s	167	Found two rightmost eigenvalues
SI A_s with $\sigma = 0$	7	Found rightmost eigenvalue
SI $A - \sigma E$ with $\sigma = 0$	300+	No results
Rightmost A_s via Algorithm 1	167	Found two rightmost eigenvalues

in the number of iterations and convergence when working with A_s directly or using Algorithm 2. Thirdly, if we apply Shift-and-Invert to the descriptor case, the process stagnates due to the fact that approximations of eigenvalues at infinity are selected (and never converge): due to round-off errors these approximations of eigenvalues at infinity appear, even when the Arnoldi process is started with a purified initial vector free of components in the direction of the corresponding eigenvectors [22,27]. These problems do not occur when working with the state space matrices.

The results for applying Arnoldi to the system with Kpss = 10 (there are no eigenvalues in the right half-plane) are shown in Table 2 and Fig. 5. We can make similar observations from Table 2, where the results are shown for a system with no eigenvalues in the right half-plane: again targeting the rightmost eigenvalues for the state space matrix A_s, whether explicitly or implicitly, works best. Furthermore, the process for the Shift-and-Invert with the descriptor matrices stagnates due to the presence of eigenvalues at infinity.

Further investigation showed that the stagnation after having found one eigenvalue was not caused by the relatively high tolerance used for checking convergence: decreasing the tolerance to $\tau = 10^{-10}$ and improving eigenvector approximations with for instance one or two refining Rayleigh Quotient Iterations did not help.

4.2. Large-scale power system model

The large system is a 1997 planning model for the Brazilian Interconnected Power System (BIPS/97). The state space realization has 1664 states, and the descriptor realization is of order 13,250 [28]. The power system stabilizer gain [10] at a critical power station was set at a very low value: Kpss = 1. The rightmost eigenvalue, associated with the major north–south low-frequency oscillation of BIPS, became then unstable. The rightmost eigenvalue was $\lambda \approx 0.031 \pm 1.1i$. The dimension of the Arnoldi basis in IRA was 20. The results are summarized in Table 3 and Fig. 6.

For this case Arnoldi applied to A_s with target rightmost does not converge within 300 restarts, which is most probably due to a number of eigenvalues clustering together [34]. It can also be observed that the Shift-and-Invert approach shows similar performance as for the small test case: it finds the rightmost eigenvalue, but then stagnates due to clustering of eigenvalues (for A_s) or hampering by eigenvalues at infinity (for the descriptor case). We also see that working with implicit solves via the descriptor matrices is much faster than working with the state space matrix A_s directly, especially for Shift-and-Invert (although it does not help in finding more eigenvalues).

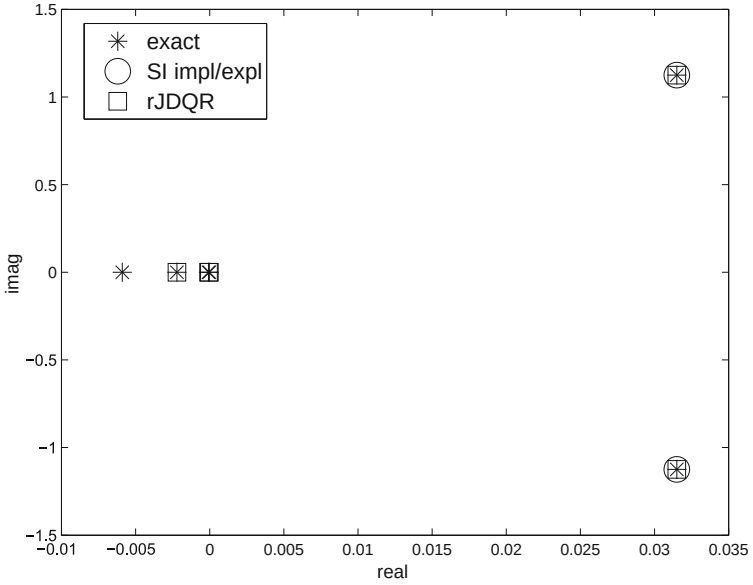

Fig. 6. Rightmost part of the spectra for Kpss = 1. Shift-and-Invert Arnoldi with target 'rightmost' converges to the rightmost eigenvalue, but fails to find the second rightmost (Arnoldi applied to the state space matrix does not converge to any eigenvalue). Jacobi–Davidson finds the three rightmost eigenvalues.

There are several ways to improve the convergence of Shift-and-Invert Arnoldi. In [6,9,27,22] techniques based on (modified) Cayley transforms are described. Here we will consider the Jacobi–Davidson method as an alternative (the approach differs slightly from the approach in [27]): a real variant [39] of JDQR is used to compute the rightmost eigenvalues of A_s. Every iteration the rightmost Ritz value was selected as shift for the next iteration. The JD process was restarted at search space dimension $j_{\max} = 30$, to a search space of dimension $j_{\min} = 10$. Note that carrying out all operations (MVs, correction equation solves) with A_s implicitly is much faster than working explicitly with A_s: 18s vs. 331s. As can be seen in Table 3 and Fig. 6, the Jacobi–Davidson method succeeds in finding the three rightmost eigenvalues.

For another example ($n_s = 1257$, $n_z = 10,482$), where several PSSs were disabled to generate three complex-conjugated eigenvalues in the right half-plane, Jacobi–Davidson with the same configuration also succeeded to find all three pairs, plus the rightmost eigenvalues in the left half-plane (cf. Table 4). The resulting spectrum is shown in Fig. 7. Shift-and-Invert Arnoldi, with target rightmost was not able to find any eigenvalues within 200 restarts. Choosing shifts close to the rightmost eigenvalues helps Arnoldi to converge, but this is impractical in general since their location is usually unknown and a fairly good eigenvalue approximation for every right half-plane eigenvalue may be needed as shift(s) to find all eigenvalues of interest.

In some sense, this comparison might seem unfair, since Jacobi–Davidson gets a new shift at every iteration, while Arnoldi has to work with the shift that was provided at the start (typically $\sigma = 0$). One could consider to restart the Arnoldi process with new shifts, based on the current Ritz values or available eigenvalues, in order to improve convergence to the rightmost eigenvalues. The currently available rightmost eigenvalues can be used to determine a Cayley transformation that transforms any remaining eigenvalues to the right of the available eigenvalues, outside the unit circle, while eigenvalues to the left are transformed inside the unit circle [27]. The Arnoldi method can then be applied to the transformed problem to find the largest eigenvalues (that correspond to eigenvalues to the right of the already found eigenvalues). This can be repeated until no new rightmost eigenvalues are found. As can be seen in Table 4, this approach is successful as well, but compared to Jacobi–Davidson it is more than 20 times slower (3 IRA restarts were needed to find the eigenvalues closest to zero in the left half-plane; 20 restarts were needed for the first Cayley transformed problem to

Table 3
Results for Implicitly Restarted Arnoldi and Jacobi–Davidson (real JDQR) applied to $A_s\mathbf{x} = \lambda\mathbf{x}$ for computing two rightmost eigenvalues (both methods with target rightmost). In this case the rightmost eigenvalue was $\lambda = 0.031 \pm 1.1i$ (cf. Fig. 6).

Target	Time (s)	# Restarts	Remarks
IRA A_s	200	300+	No eigenvalues found
IRA SI A_s ($\sigma = 0$)	2000+	100+	One rightmost, stagnation
IRA SI $A - \sigma E$ ($\sigma = 0$)	100	300+	One rightmost
IRA A_s via Algorithm 1	100	300+	No eigenvalues found
IRA SI via Algorithm 2	44	30	One rightmost, stagnation
rJDQR(A_s)	331	3	Three rightmost found
rJDQR(A_s) (Algorithms 1 and 2)	18	3	Three rightmost found

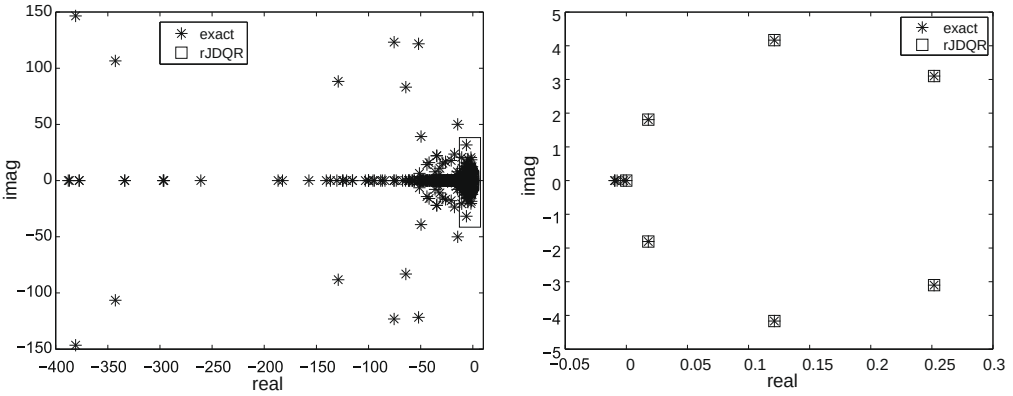

Fig. 7. Complete spectrum (left, two more leftmost eigenvalues not shown) and rightmost part of the spectrum (within boxed area) for power system without PSSs. Shift-and-Invert Arnoldi with target 'rightmost' does not converge to any eigenvalue within 200 restarts. Jacobi–Davidson finds all three eigenvalues in the right half-plane, as well as the two rightmost eigenvalues in the left half-plane.

find the leftmost eigenvalue in the right half-plane; finally, 192 restarts were needed for next Cayley transformed problem to find the remaining rightmost eigenvalues). The slow convergence can be explained by the fact that the eigenvalues of the Cayley transformed problems tend to have magnitude (very) close to 1 (even if the (real) shifts are close to the real part of the eigenvalues, the imaginary parts cause the transformed eigenvalues to have magnitude close to 1).

Based on these results and other results not reported here the Jacobi–Davidson method, provided directly applied to the state space matrices (with implicit computations for efficiency), seems to be better able to compute the rightmost eigenvalues both in the right and left half-plane. Arnoldi with Shift-and-Invert does compute the rightmost eigenvalues in the right half-plane, but fails to compute the rightmost eigenvalues in the left half-plane, due to the fact that the Shift-and-Invert transformation changes the relative location of the eigenvalues. As described, there exist advanced Cayley shift selection strategies to deal with this [6,27], but the Jacobi–Davidson approach is more effective and simpler in the sense that the natural choice of selecting rightmost Ritz values is enough in practice.

4.3. Other applications

The principle of implicitly computing in state space by using the sparse descriptor format is also described in [12,15] for solving large-scale Lyapunov equations. The advantages for this approach are that linear system solves are much cheaper with sparse descriptor matrices, that projections on the finite eigenspaces are not needed (contrary to when computing solutions of generalized Lyapunov

Table 4
Results for Implicitly Restarted Arnoldi and Jacobi–Davidson (real JDQR) applied to $A_s\mathbf{x} = \lambda\mathbf{x}$ for computing five rightmost eigenvalues. In this case there were three eigenvalues in the right half-plane (cf. Fig. 7).

Target	Time (s)	# Restarts	Remarks
IRA SI via Algorithm 2	200+	100+	No eigenvalues found
rJDQR(A_s) (Algorithms 1 and 2)	15	90	Five rightmost found
IRA Cayley via Algorithm 2	337	3+20+192	Five rightmost found

equations corresponding to descriptor systems [35,36]), and that the reduced-order model can be constructed from the (projected) state space matrices. The latter advantage one also has when the Arnoldi basis constructed in the previous examples is used for projecting the state space matrices to obtain a reduced-order model: the reduced-order model will be free of spurious approximations of poles at infinity.

5. Conclusions

In this paper, we have shown how structure in electrical and power system problems can be exploited to make numerical methods more robust and efficient. The practical need for structure exploitation comes from the fact that, with increasing complexity of systems such as electrical circuits and power systems, direct application of established numerical methods may lead to inaccurate results and/or inefficient computations. We have described how, by making use of the structure of the underlying problem, the performance of numerical algorithms can be improved considerably, without changing intrinsic properties of the algorithms.

Structure exploitation requires that the structure of the problem is known. In many practical applications, such as design and simulation of electrical circuits and power systems, this is the case. An important concept is that one should compute with sparse descriptor matrices as much as possible, and refrain from working with dense state space matrices. Dense state space matrix formulations might have the advantage of having much smaller dimensions, but for present systems the number of states easily exceeds several thousands and hence established methods such as the QR and QZ method for eigenvalue problems, are no longer a practical choice. Besides that, efficient algorithms for descriptor matrix formulations have been developed recently.

In some cases algorithms originally designed for state space systems can still be used. The key insight (and requirement) is that operations with state space matrices can be done implicitly with the sparse descriptor matrices. Typical operations needed by iterative algorithms are matrix–vector multiplications and solves with (shifted) state space matrices, and these operations can be implemented implicitly with descriptor matrices. The resulting approach combines the advantages of the robustness of these algorithms with efficiency of sparse matrix computations. We have shown how this can be done for eigenvalue problems, solution of Lyapunov equations, and model order reduction.

Acknowledgments

The authors would like to thank the anonymous referees for valuable comments that helped us to improve the presentation of the paper. This paper is inspired by discussions with, and lectures by, Henk van der Vorst, where the importance of the underlying mathematical and physical properties of problems was emphasized.

References

[1] Hsim. <http://www.synopsys.com/products/mixedsignal/hsim/hsim.html>.
[2] P.R. Amestoy, T.A. Davis, I.S. Duff, An approximate minimum degree ordering algorithm, SIAM J. Matrix Anal. Appl. 17 (4) (1996) 886–905.
[3] A.C. Antoulas, Approximation of Large-Scale Dynamical Systems, SIAM, 2005.
[4] U. Baur, Control-Oriented Model Reduction for Parabolic Systems, Ph.D. Thesis, Technische Universität, Berlin, 2008.

[5] P. Benner, V. Mehrmann, D. Sorensen (Eds.), Dimension Reduction of Large-Scale Systems, Lecture Notes in Computational Science and Engineering, vol. 45, Springer, 2005.
[6] L.H. Bezerra, C. Tomei, Spectral transformation algorithms for computing unstable modes, Comput. Appl. Math. 18 (1) (1999) 1–15.
[7] C.W. Bomhof, Jacobi–Davidson methods for eigenvalue problems in pole zero analysis, Nat. Lab. Unclassified Report 012/97, Philips Electronics NV, 1997.
[8] P.G. Ciarlet, W.H.A. Schilders, E.J.W. ter Maten (Eds.), Numerical Methods in Electromagnetics, Handbook of Numerical Analysis, vol. 13, Elsevier, 2005.
[9] K.A. Cliffe, T.J. Garratt, A. Spence, Eigenvalues of block matrices arising from problems in fluid mechanics, SIAM J. Matrix Anal. Appl. 15 (4) (1994) 1310–1318.
[10] J.C.R. Ferraz, N. Martins, G.N. Taranto, Coordinated stabilizer tuning in large power systems considering multiple operating conditions, in: IEEE/PES General Meeting, June 2007, No. 07GM0304.
[11] D.R. Fokkema, G.L.G. Sleijpen, H.A. van der Vorst, Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils, SIAM J. Sci. Comput. 20 (1) (1998) 94–125.
[12] F.D. Freitas, J. Rommes, N. Martins, Gramian-based reduction method applied to large sparse power system descriptor models, IEEE Trans. Power Syst. 23 (3) (2008) 1258–1270.
[13] R.W. Freund, SPRIM: Structure-preserving reduced-order interconnect macromodeling, in: Technical Digest of the 2004 IEEE/ACM International Conference on CAD, 2004, pp. 80–87.
[14] G.H. Golub, C.F. van Loan, Matrix Computations, third ed., John Hopkins University Press, 1996.
[15] M. Heinkenschloss, D.C. Sorensen, K. Sun, Balanced truncation model reduction for a class of descriptor systems with application to the Oseen equations, SIAM J. Sci. Comput. 30 (2) (2008) 1038–1063.
[16] P. Kundur, Power System Control and Stability, McGraw-Hill, 1994.
[17] R.B. Lehoucq, D.C. Sorensen, Deflation techniques within an implicitly restarted Arnoldi iteration, SIAM J. Matrix Anal. Appl. 17 (1996) 789–821.
[18] R.H. Lehoucq, D.C. Sorensen, C. Yang, ARPACK SOFTWARE. <http://www.caam.rice.edu/software/ARPACK/>.
[19] J.-R. Li, J. White, Low rank solution of Lyapunov equations, SIAM J. Matrix Anal. Appl. 24 (1) (2002) 260–280.
[20] N. Martins, Efficient eigenvalue and frequency response methods applied to power system small-signal stability studies, IEEE Trans. Power Syst. 1 (1) (1986) 217–226.
[21] N. Martins, L.T.G. Lima, H.J.C.P. Pinto, Computing dominant poles of power system transfer functions, IEEE Trans. Power Syst. 11 (1) (1996) 162–170.
[22] K. Meerbergen, A. Spence, Implicitly restarted Arnoldi with purification for the shift–invert transformation, Math. Comp. 66 (218) (1997) 667–689.
[23] K. Meerbergen, A. Spence, D. Roose, Shift–invert and Cayley transforms for detection of rightmost eigenvalues of nonsymmetric matrices, BIT 34 (3) (1994) 409–423.
[24] B.C. Moore, Principal component analysis in linear systems: controllability, observability and model reduction, IEEE Trans. Automat. Control 26 (1) (1981) 17–32.
[25] T. Penzl, Algorithms for model reduction of large dynamical systems, Linear Algebra Appl. 415 (2–3) (2006) 322–343.
[26] J.R. Phillips, L.M. Silveira, Poor man's TBR: a simple model reduction scheme, IEEE Trans. CAD Circ. Syst. 24 (1) (2005) 283–288.
[27] J. Rommes, Arnoldi and Jacobi–Davidson methods for generalized eigenvalue problems $Ax = \lambda Bx$ with singular B, Math. Comput. 77 (262) (2008) 995–1015.
[28] J. Rommes, N. Martins, Computing large-scale system eigenvalues most sensitive to parameter changes, with applications to power system small-signal stability, IEEE Trans. Power Syst. 23 (4) (2008) 434–442.
[29] J. Rommes, N. Martins, Efficient computation of transfer function dominant poles of large second-order dynamical systems, SIAM J. Sci. Comput. 30 (4) (2008) 2137–2157.
[30] J. Rommes, G.L.G. Sleijpen, Convergence of the dominant pole algorithm and Rayleigh quotient iteration, SIAM J. Matrix Anal. Appl. 30 (1) (2008) 346–363.
[31] W.H.A. Schilders, H.A. van der Vorst, J. Rommes (Eds.), Model Order Reduction: Theory, Research Aspects and Applications, Mathematics in Industry, vol. 13, Springer, 2008
[32] G.L.G. Sleijpen, H.A. van der Vorst, A Jacobi–Davidson iteration method for linear eigenvalue problems, SIAM J. Matrix Anal. Appl. 17 (2) (1996) 401–425.
[33] G.L.G. Sleijpen, H.A. van der Vorst, M. van Gijzen, Quadratic eigenproblems are no problem, SIAM News 29 (7) (1996) 8–9.
[34] D.C. Sorensen, Implicit application of polynomial filters in a k-step Arnoldi method, SIAM J. Matrix Anal. Appl. 13 (1992) 357–385.
[35] T. Stykel, Analysis and Numerical Solution of Generalized Lyapunov Equations, Ph.D. Thesis, Technischen Universität Berlin, 2002.
[36] T. Stykel, Gramian based model reduction for descriptor systems, Math. Control Signals Syst. 16 (2004) 297–319.
[37] The Mathworks, Inc. Matlab.
[38] F. Tisseur, K. Meerbergen, The quadratic eigenvalue problem, SIAM Rev. 43 (2) (2001) 235–286.
[39] T.L. van Noorden, J. Rommes, Computing a partial generalized real Schur form using the Jacobi–Davidson method, Numer. Linear Algebra Appl. 14 (3) (2007) 197–215.
[40] F. Yang, X. Zeng, Y. Su, D. Zhou, RLC equivalent circuit synthesis method for structure-preserved reduced-order model of interconnect in VLSI, Commun. Comput. Phys. 3 (2) (2008) 376–396.